Unexpected but Informative: Frontal Positivity Elicited by Confusing Program Code Mirrors Natural Language Processing

Annabelle Bergum[1+], Anna-Maria Maurer[1+] (maureran@cs.uni-saarland.de), Norman Peitek[1], Regine Bader[2], Axel Mecklinger[2], Vera Demberg[1,3], Janet Siegmund[4], and Sven Apel[1]

[1]Computer Science, Saarland University, [2] Psychology, Saarland University, [3] Language Science and Technology, Saarland University, [4] Computer Science, University of Technology Chemnitz, [+]These authors contributed equally to this work

Insights from natural language processing provide a powerful framework for understanding how the human brain comprehends program code as a structured representational system. While programming languages and natural languages differ in purpose and form, both require incremental parsing, prediction, and integration of incoming information.

In this study, we investigated the neurophysiological responses underlying the comprehension of program code and drew parallels to natural language processing [3]. As a phenomenon, we analyzed “atoms of confusion,” code patterns that are syntactically valid but established as cognitively misleading for human programmers [4]. We contrasted them against equivalent, non-confusing code patterns. Listings 1–6 show some examples. During naturalistic code reading, we recorded synchronized EEG and eye-tracking data from 24 programmers on 36 confusing and 36 non-confusing code snippets during output determination. We captured the neurophysiological responses in form of fixation-related potentials (FRPs) at the first fixation in the region of interest (ROI) containing the atom or the non-confusing equivalent (-100–0 ms baseline, 0–1 000 ms). To analyze the FRPs, we performed a paired cluster-permutation test in the interval of 100–1 000 ms.

Behavioral results validated the presence of confusion based on duration, correctness and difficulty. The first fixation in ROI occurred slightly later and lasted longer for confusing code (at 3.8±3.4 s for 0.8 ± 0.9 s) compared to non-confusing stimuli (at 3.2 ± 3.0 s for 0.6 ± 0.6 s). Our FRP results reveal a late frontal positivity occurring approximately 400–800 ms after fixation onset on the ROI [3], as shown in Figure 1. This neural signature closely resembles the late frontal positivity component observed in natural language processing during the integration of unexpected yet plausible input within an evolving mental representation [1].

Within frameworks of language processing [1,2], such late positivities have been linked to processes the resolution of conflicts between the situation model and unpredicted information. We propose that atoms of confusion in program code trigger such processes, as they introduce locally coherent but globally misleading structures that require revision during comprehension. The observed effect therefore reflects the effortful integration of these patterns into a coherent situation model. Accordingly, the comprehension of program code appears to recruit domain-general mechanisms for handling prediction violations and updating situation models.

References

[1] Gina Kuperberg, Trevor Brothers, Edward Wlotko. A Tale of Two Positivities and the N400: Distinct Neural Signatures Are Evoked by Confirmed and Violated Predictions at Different Levels of Representation. Journal Cognitive Neuroscience 32(1):12-35, 2020

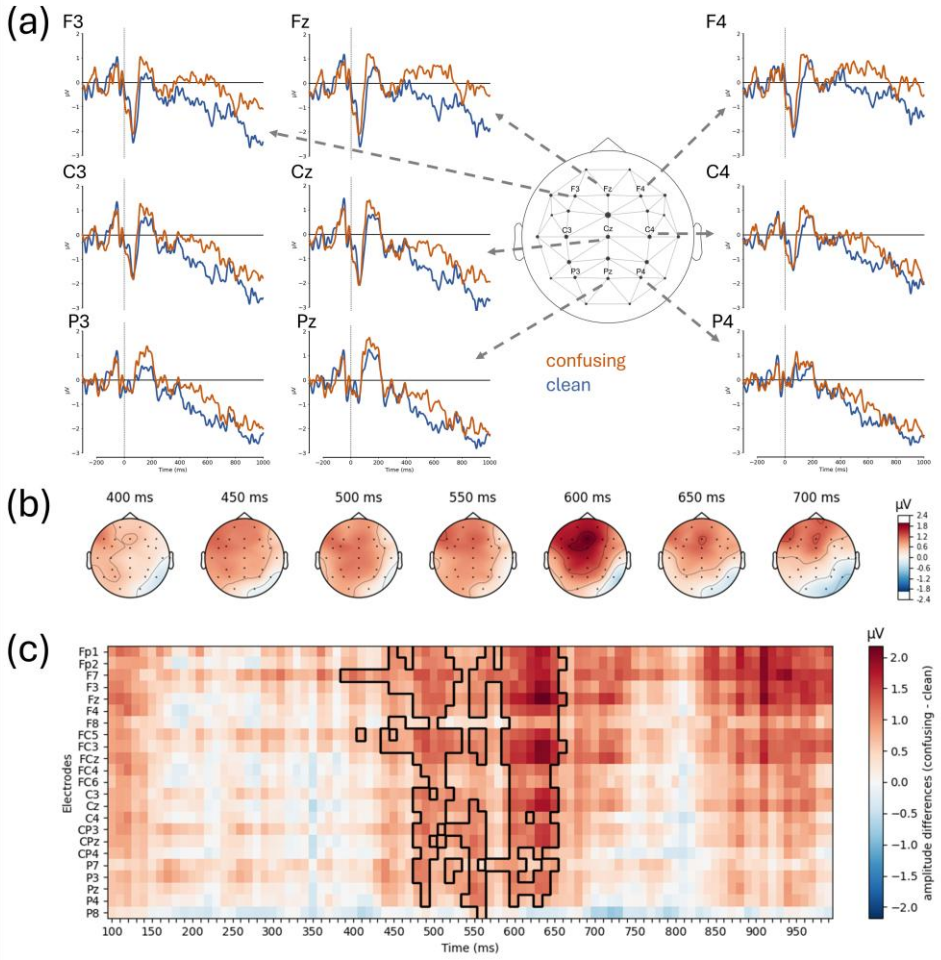
[2] Angela Friederici. The Brain Basis of Language Processing: From Structure to Function, Physiological Reviews 91:4, 1357-1392, 2011

[3] Annabelle Bergum, Anna-Maria Maurer, Norman Peitek, Regine Bader, Axel Mecklinger, Vera Demberg, Janet Siegmund, Sven Apel. Fixation-related potentials reveal that confusing program code elicits a late frontal positivity. *Sci Rep* **16**, 16833 (2026)

[4] Dan Gopstein, Jake Iannacone, Yu Yan, Lois DeLong, Yanyan Zhuang, Martin K.-C. Yeh, Justin Cappos: Understanding misunderstandings in source code. Joint Meeting on Foundations of Software Engineering, 2017, 129-139,

Figures and Tables

Figure 1: Results of the FRP analysis. Part (a) shows the FRP elicited by confusing (orange) and clean (blue) program code at nine scalp electrodes. Part (b) portrays the topographic distribution of the amplitude difference in consecutive 50 ms time intervals. Part (c) shows the amplitude differences between confusing and clean program code for all electrodes between 100 and 1 000 ms after fixation onset. The significant cluster is surrounded in black.



Optional Supplemental Information

Atoms of confusion are small patterns of code that have been shown empirically to confuse programmers. In those patterns, indentations, operator precedence, type conversions, encodings, or other features of the programming language are used in an unintuitive manner. https://atomsofconfusion.com/

Confusing code containing an atom of confusion	Equivalent clean code without an atom of confusion

`1 boolean V1 = true;`
`2 boolean V2 = false;`
`3 int R = 3;`
`4 if (V1)`
`5  if (V2)`
`6  R = R + 2;`
`7 else`
`8  R = R + 4;`
 	
Listing 1. Indentation of the `else` block makes it difficult to match it to the correct `if` statement (
	`1 boolean V1 = true;`
	`2 boolean V2 = false;`
	`3 int R = 3;`
	`4 if (V1)`
	`5  if (V2)`
	`6  R = R + 2;`
	`7  else`
	`8  R = R + 4;`
	
	Listing 2. `else` block is unambiguously indented to make it easy to match it to the second `if` statement (

`1 boolean V1 = false;`
`2 boolean V2 = true;`
`3 boolean V3 = true;`
`4 boolean R = (V1 && V2 || V3);`
 	
Listing 3. At the operator `||` it is unclear whether the `&&` or the `||` operator is executed first (
	`1 boolean V1 = false;`
	`2 boolean V2 = true;`
	`3 boolean V3 = true;`
	`4 boolean R = ((V1 && V2) || V3);`
	
	Listing 4. Parenthesis render the order of the operators `||` and `&&` unambiguous (

`1 int V1 = 2;`
`2 int R = 3 + V1++;`
 	
Listing 5. At the operator `++` it is unclear whether the `+` or the `++` operator is executed first
	`1 int V1 = 2;`
	`2 int R = 3 + V1;`
	`3 V1++;`
	
	Listing 6. The extraction of `V1++` into a separate statement render the order of the operators `+` and `++` unambiguous

Natural language examples for late frontal positivity:

He bought her a pearl necklace for her collection.	He bought her a pearl necklace for her birthday.
The word “collection” does not fit the expectation of the context created by the prior sentence, even though it is semantically plausible. This sentence results in a late frontal positivity.	The sentence context suggests, shown by cloze tests, that the next expected word is “birthday” with a high probability. This sentence does not result in a late frontal positivity.